AFRL-HE-WP-TR-2006-0064

AIR FORCE RESEARCH LABORATORY

Quantifying the Impacts of Improvements to Prognostic and Diagnostic Capabilities

C. Richard Cassady
Edward A. Pohl
Jason Honeycutt
Letitia Pohl
Mauricio Carrasco

University of Arkansas
Department of Industrial Engineering
4207 Bell Engineering Center
Fayetteville, AR 72701

January 2005

Interim Report for November 2002 to January 2005

Human Effectiveness Directorate
Warfighter Readiness Research Division
Logistics Readiness Branch
2698 G Street
Wright-Patterson AFB OH 45433-7604

TECHNICAL REVIEW AND APPROVAL

AFRL-HE-WP-TR-2006-0064

This technical report has been reviewed and is approved for publication.

FOR THE COMMANDER

//SIGNED//

DANIEL R. WALKER, Colonel, USAF
Chief, Warfighter Readiness Research Division
Human Effectiveness Directorate

REPORT DOCUMENTATION PAGE	Form Approved OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)* January 2005	2. REPORT TYPE Interim	3. DATES COVERED *(From - To)* November 2002 - January 2005
4. TITLE AND SUBTITLE Quantifying the Impacts of Improvements to Prognostic and Diagnostic Capabilities		5a. CONTRACT NUMBER F33615-99-D-6001
		5b. GRANT NUMBER
		5c. PROGRAM ELEMENT NUMBER 63231F
6. AUTHOR(S) C. Richard Cassady, Edward A. Pohl, Jason Honeycutt, Letitia Pohl, Mauricio Carrasco		5d. PROJECT NUMBER
		5e. TASK NUMBER
		5f. WORK UNIT NUMBER 49230208
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) University of Arkansas, Department of Industrial Engineering, 4207 Bell Engineering Center, Fayetteville, AR 72701		8. PERFORMING ORGANIZATION REPORT NUMBER MM0303
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Air Force Materiel Command, Air Force Research Laboratory, Human Effectiveness Directorate, Warfighter Readiness Research Division, Logistics Readiness Branch, Wright-Patterson AFB OH 45433-7604		10. SPONSOR/MONITOR'S ACRONYM(S) AFRL/HEAL
		11. SPONSOR/MONITOR'S REPORT NUMBER(S) AFRL-HE-WP-TR-2006-0064

12. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES
DO #26, Task 3 Cleared as AFRL/WS-06-1580, 22 Jun 06.

14. ABSTRACT
A key challenge faced by USAF maintenance personnel is the imperfection of aircraft diagnostic and prognostic capabilities. These imperfections include an inability to pinpoint failed components, the incorrect identification of failed components, and "cannot-duplicate" errors between the flightline and the depot. In addition to lower morale in maintenance personnel, these imperfections lead to increased delays in returning aircraft mission capable and excessive requirements for spare parts in the supply chain. Unfortunately, it is difficult to assess the impact of improvements to diagnostic and prognostic capabilities. The objective of this project is to develop a methodology based in mathematical modeling for analyzing these impacts. Specifically, we address the following questions: (1) What impact do diagnostic and prognostic errors have on fleet performance? (2) Given a specific investment in diagnostic improvements, what will the impact be on fleet performance? (3) Given a limited budget for diagnostic improvements, how should the funds be allocated to optimize fleet performance? The activities required to achieve the objective of this project are applied to hypothetical systems that possess several key characteristics similar to systems utilized by the US Air Force.

15. SUBJECT TERMS
Aircraft Maintenance, Maintenance Technicians, Diagnostic Tool, Prognostic Tool, Mathematical Modeling, Artificial Neural Networks

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON Cheryl L. Batchelor
a. REPORT UNCLASSIFIED	b. ABSTRACT UNCLASSIFIED	c. THIS PAGE UNCLASSIFIED	SAR	40	19b. TELEPHONE NUMBER *(include area code)*

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. 239.18

THIS PAGE LEFT INTENTIONALLY BLANK

Executive Summary

A key challenge faced by USAF maintenance personnel is the imperfection of aircraft diagnostic and prognostic capabilities. These imperfections include an inability to pinpoint failed components, the incorrect identification of failed components, and "cannot-duplicate" errors between the flightline and the depot. In addition to lower morale in maintenance personnel, these imperfections lead to increased delays in returning aircraft mission capable and excessive requirements for spare parts in the supply chain. Unfortunately, it is difficult to assess the impact of improvements to diagnostic and prognostic capabilities. The objective of this project is to develop a methodology based in mathematical modeling for analyzing these impacts. Specifically, we address the following questions: (1) What impact do diagnostic and prognostic errors have on fleet performance? (2) Given a specific investment in diagnostic improvements, what will the impact be on fleet performance? (3) Given a limited budget for diagnostic improvements, how should the funds be allocated to optimize fleet performance?

The activities required to achieve the objective of this project are applied to hypothetical systems that possess several key characteristics similar to systems utilized by the US Air Force. In the first phase of the project, we create a set of mathematical and logical modeling tools related to diagnostic errors. We establish a modeling framework for a single system, and we describe simulation modeling efforts for evaluating the readiness of a single system and a set of systems under diagnostic errors. In addition, we present an optimization model for allocation a diagnostic improvement budget.

In the second phase of this research, we use a simulation modeling environment to compare prognostics to scheduled maintenance. As a first step, we establish a modeling framework and create the simulation model required to accompany this framework. Then, we use a numerical example to demonstrate the capability of the model. Specifically, we show that prognostics can be an effective tool in some cases (even in the presence of significant prognostics errors) and a very ineffective tool in other cases. In some cases, prognostic errors can make a situation worse than "run to failure".

Table of Contents

Tables

Figures

1. Introduction

A key challenge faced by USAF maintenance personnel is the imperfection of aircraft diagnostic and prognostic capabilities. These imperfections include an inability to pinpoint failed components, the incorrect identification of failed components, and "cannot-duplicate" errors between the flightline and the depot. In addition to lower morale in maintenance personnel, these imperfections lead to increased delays in returning aircraft to the fleet and excessive requirements for spare parts in the supply chain. Unfortunately, it is difficult to assess the impact of improvements to diagnostic and prognostic capabilities. The objective of this project is to develop a methodology based in mathematical modeling for analyzing these impacts. Specifically, we intend to address the following questions: (1) What impact do diagnostic and prognostic errors have on fleet performance? (2) Given a specific investment in diagnostic improvements, what will the impact be on fleet performance? (3) Given a limited budget for diagnostic improvements, how should the funds be allocated to optimize fleet performance?

The activities required to achieve the objective of this project are applied to a hypothetical fleet of systems. However, the hypothetical systems possess several key characteristics similar to systems utilized by the US Air Force (and other organizations). We define the system structure and the reliability and maintainability characteristics of each component in the system. In addition, we identify the characteristics of the diagnostic and prognostic tools applied to the system. This identification includes a precise description of the potential for imperfections in diagnostic and prognostic information. We develop a set of mathematical and logical models which can be used to measure the performance of the fleet with and without diagnostic and prognostic errors. This permits the assessment of the impact of diagnostic and prognostic imperfections on fleet performance. Using these models, we explore the impact of specific investments in diagnostic and prognostic tools. This permits evaluation of the cost-effectiveness of potential diagnostic and prognostic improvement actions. In addition, we incorporate the models into a decision-support environment that is designed to allocate investments in diagnostic improvements. This permits the decision-maker to fund cost-effective diagnostic improvement efforts that optimize fleet performance.

The remainder of this report is summarized as follows. In Section 2, we summarize the relevant research literature. Section 3 presents the development a simulation-based framework for assessing the impact of diagnostics errors and associated improvements efforts. In this section, we also explore an approach for optimizing the allocation of a budget designated for diagnostic improvement efforts. Section 4 describes a simulation-based effort for capturing the impacts of errors in prognostic capabilities.

2. Research Literature Review

The purpose of this literature review is to identify existing mathematical modeling techniques used in the area of diagnostics. Diagnostics is the first step in the repair process and involves identifying the cause of a failure. Typically, the goal is to isolate the failure to a faulty module and/or component, and this is done based on system observations and available test data. Often, the determination that a failure has occurred is one step (e.g., the failure of a built-in test) and the isolation of that failure is a second step. In other applications, however, failure detection and isolation are not separable. For example, the diagnostic problem may be formulated as a classification problem, where the system state is classified as either normal operation, or as one of several possible failure modes [6].

We are interested in models that can be used to evaluate the effectiveness of a system's diagnostics, not necessarily modeling techniques that are used in the design of a diagnostics system. The latter type may require detailed system-specific information, which is not readily available to us. However, these techniques can apply to our research and are therefore summarized, as well. We are particularly interested in techniques that take into consideration imperfect test results, which introduce uncertainty into the diagnosis. The two main types of test error include (1) the test indicates a pass, when in fact, the unit under test has failed, and (2) the test indicates a fail, when the unit is working properly (a false alarm).

Fault diagnosis in large-scale systems has been a major research area for several decades and there is considerable literature available. The inter-disciplinary problem of diagnostics is a concern in all stages of the product life cycle, but particularly during manufacturing and field maintenance [3]. It has therefore been approached from the perspective of the electronics design engineer, the diagnostics software developer, the reliability engineer, and others. Many of these techniques require specific information about the system design, and in fact, the models may be constructed during the design phase. Because the diagnostics process has traditionally been dependent on human involvement, the development of automated diagnostics systems has also frequently relied on artificial intelligence (AI) techniques.

There are parallels between system fault diagnosis and the diagnosis of diseases in the medical field. In both, there is a search for the cause of a failure (or illness) based on symptoms, and often, there is a desire to develop an optimal test sequence for a given symptom. Some mathematical tools are used similarly in both areas, but for the purpose of this review, medical diagnosis techniques are not included.

A common and well established approach to diagnostics is the Bayesian process. Bayesian inference can be used to determine the probability that a diagnosis is correct. However, it has the disadvantage of requiring a priori probability distributions, which may not always be available [12].

Fenton [3], in his review of AI approaches to diagnostics, states that model-based diagnosis involves using the model to predict faults from observations and information on the real device or system. He identifies four types of models and provides numerous references as examples of their application: fault models (or fault dictionaries), causal models, models based on structure and behavior, and diagnostic inference models.

Fault models anticipate the types of faults that may occur and only model those. Each fault type is inserted into the system and the system behavior is monitored. From this, a list of fault/symptom pairs or fault dictionary is produced. This method has been used primarily for digital circuit diagnosis. These models are unable to handle unanticipated faults.

A causal model is a directed graph, where nodes represent symptoms and faults, and the links represent the relationships between them. The strength of each link is often defined using a numerical weight or probability. The fault hypotheses are ranked or eliminated using Bayesian techniques. Bayesian networks are a variation on this approach.

Models based on structure and behavior require detailed information on the system components, their interconnections, and the behavior pattern for each component. If the system outputs do not agree with the model for a given set of inputs, then it is assumed a discrepancy has occurred and diagnosis must be performed to find the defective component. This type of model can theoretically diagnose any fault type, which overcomes the disadvantage of a fault model, which cannot detect unanticipated errors [3]. The diagnosis then requires a series of hypothesis tests to determine if each suspect component can account for all the observations. Diagnosis may require further testing, and there are techniques to determine the best test sequence.

Diagnostic inference models represent the problem as a flow of diagnostic information. The model consists of two basic elements: tests and conclusions. Tests may be any source of diagnostic information, including observable symptoms, logistics history and results from diagnostic tests. Conclusions typically represent faults or units to replace. The dependency relationship between tests and conclusions is represented using a directed graph. Test sequencing is optimized using algorithms based on maximum test information gain. Diagnostic inference combines information from multiple tests using several logical and statistical inference techniques, including a modified form of Dempster-Shafer (D-S) evidential reasoning which compensates for disappearing uncertainty in the face of conflict. Conflicts are caused by test error, multiple faults, or incomplete or inaccurate models [3]. The Dempster-Shafer approach is also addressed in [12].

In [6], the types of models used in diagnostics are identified as physical models, reliability models, machine learning models, and dependency models. Physical models are based on the natural laws governing system operation, e.g., material properties (solid, liquid, gas), finite-element models, thermodynamics, etc. A physics-based failure model usually needs to be built for each failure mode, and requires intricate knowledge by area experts. Reliability modeling requires knowledge of the system structure and failure probability distributions. Machine learning models are purely data dependent models and require historical training data. Neural networks are the primary example of machine learning models. Dependency models capture cause and effect relationships. An example of a failure dependency model is provided in [6].

Deb *et al.* [2] describe four modeling techniques for diagnosing faults in complex systems: quantitative, qualitative, structural and dependency. Quantitative models require highly detailed system information and provide an exact simulation of the system. Qualitative models are simplified quantitative models. Structural models represent the connectivity and failure propagation direction in the form of a directed graph. An example is found in [4], where a directed graph is used to represent the propagation of a fault through the system. Each node represents a unit (or its failure mode) and a link between two nodes indicates that a fault can propagate from one to the other. Dependency models (similarly defined in [6]) represent the cause-effect relationships in the form of a directed graph, and can deviate significantly from structural models. According to [2], dependency modeling, which is also referred to elsewhere as inference modeling, is the primary modeling technique employed in testability analysis tools.

This paper points out the disadvantages of each of these modeling techniques, the expense involved in developing them and notes that they have been successfully applied, although at a steep price, to many defense projects. In the author's opinion (dated 1995) they are not cost effective.

Fenton [3] summarizes the use of fuzzy logic and artificial neural networks in diagnostics. Fuzzy logic can be used to represent uncertainty and inaccurate data in a diagnostics environment - approximations rather than exact measurements. It can also be used to incorporate qualitative judgments from experts into an automated diagnostics system. In traditional sets, membership is either true (1) or false (0), and there is no concept of partial membership. In fuzzy sets, partial membership is allowed, so membership is represented by a value between 0 and 1. Fuzzy logic is typically combined with other modeling approaches. One such application is found in [12] and several more are identified in [3].

Artificial neural networks (ANNs) are used for a variety of applications, including diagnostics. ANNs are basically directed graphs with nodes, or neurons, connected by weighted links. Each link has an associated weight, which typically multiplies the signal transmitted along that link. Each neuron applies an activation function to its net input (sum of weighted input signals) to determine its output signal. The net can be single layer (containing only a set of input units and a set of output units, with a single set of weighted links), or more commonly, multilayer (one or more layers of nodes between the input and output units). The process of establishing the weights for each link is called training. The neural net is "trained" with data to perform a function. In the case of diagnostics, the input data may be the results of diagnostic tests and the output could be an indication of which subsystem has failed. An ANN is characterized by its structure of nodes and links, method of training, and activation function.

In [12], methods to combine system information (such as test results) to improve the confidence and accuracy of diagnostics are examined. One of the data fusion approaches proposed uses neural networks. An example is given using engine test cell data, where the output is a determination of the validity of the sensor signals, and at times, diagnosis of a sensor fault. In [15], a neural network is presented that attempts to shrink the confidence bounds around failure prediction. In [16], a self-organizing feature map (SOM) neural network, combined with fuzzy logic, is implemented. The types of neural networks most commonly used in diagnostics are multilayer, feed-forward networks with back-propagation training (see [5], [7], and [14]). Fenton [3] states that ANNs are most useful for their ability to recognize patterns and have shown promise in application where noise and error is present. Mather *et al.* [6] acknowledge that neural nets are useful for modeling phenomena that are hard to model using parametric/analytical equations. However, they are difficult to validate and do not enhance the basic understanding of the system under study.

A large portion of the literature in the diagnostics area is concerned with test sequencing. The generalized problem attempts to find the optimal test sequence which minimizes the expected cost of diagnosis (or time to test), given the test outcomes, test costs and failure probabilities. In [10], the problem is constructed as a perfectly observed Markov decision problem, where the solution is a deterministic binary decision tree. Type 1 repair (module is repaired after complete diagnosis) and type 2 repair (module is repaired after partial diagnosis) are considered, and dynamic programming recursion is used. Test sequencing algorithms based on information theory and heuristic search are developed in [9]. This problem is explored further by considering unreliable tests [11], partial and imperfect tests [17], and multiple faults [13]. In [8], the test sequencing problem is addressed for both perfect and imperfect testing. Both diagnostic error types discussed above (false-positives and false-negatives) are considered for sequential testing.

Finally, a method for evaluating the performance of automatic diagnostic systems is presented in [1]. Three measures of effectiveness for a diagnostics system are defined, which include the false positive and false negative errors previously mentioned, plus a third measure defined as false alarm correction. The false alarm correction measures the ability of the diagnostics system to correct its actions after indicating a false alarm. For the purpose of comparing various diagnostics systems, the paper develops a method for evaluating the life cycle cost of a diagnostics system, based on the three measures of effectiveness.

3. The Impact of Diagnostic Errors

In this section, we summarize a set of mathematical and logical modeling tools related to diagnostic errors. First, we establish a modeling framework for a single system. Then, we describe simulation modeling efforts for evaluating the readiness of a single system and a set of systems under diagnostic errors. Finally, we present an optimization model for allocation a diagnostic improvement budget.

3.1 System Characteristics

Consider a set of q independent and identical systems such that each subsystem is comprised of m independent subsystems. Each system is intended to operate until one of the subsystems fails. Upon failure, the system is immediately routed to the set's base of operations for maintenance. Upon the system's arrival for maintenance, technicians are assumed to know that exactly one subsystem is failed. Note that the other subsystems cannot fail while the failed system is at the base.

Each subsystem in each system has a constant failure rate during operation. Let λ_j denote the failure rate of a type j subsystem, $j = 1, 2, \dots, m$. This definition and the system structure implies that the failure rate of a single system is given by

$$\lambda = \sum_{j=1}^{m} \lambda_j \tag{3.1}$$

Furthermore, the time to system failure is an exponential random variable having rate λ, and the probability that system failure is caused by the failure of subsystem j is given by

$$\rho_j = \frac{\lambda_j}{\lambda} \tag{3.2}$$

$j = 1, 2, \dots, m$. Note that the subsystems are numbered such that

$$\rho_1 \geq \rho_2 \geq \cdots \geq \rho_m \tag{3.3}$$

3.2 Diagnostics and Their Capabilities

Upon failure, subsystems are tested sequentially until a failure is detected. Let X_j denote the time required to test subsystem j. Let C_j denote the event that subsystem j is failed, and let D_j denote the event that the diagnosis of subsystem j indicates a failure, $j = 1, 2, \ldots, m$. Note that the m diagnostics are independent of all previous and future tests.

Each of the m diagnostics is subject to two types of errors: Type I errors (false positives) and Type II errors (false negatives). For subsystem j, the probability of a Type I error is given by

$$\alpha_j = \Pr\left(D_j \middle| C_j^c\right) \tag{3.4}$$

where C_j^c denotes the complement of C_j (subsystem j is not failed), and the probability of a Type II error is given by

$$\beta_j = \Pr\left(D_j^c \middle| C_j\right) \tag{3.5}$$

where D_j^c denotes the complement of D_j (diagnostic j does not indicate failure), $j = 1, 2, \ldots, m$. Because of these errors, the probability of correct diagnosis during the first testing pass is given by

$$s_c = \sum_{j=1}^{m} \left[(1-\beta_j) \prod_{l<j} (1-\alpha_l) \right] \rho_j \tag{3.6}$$

and the probability of "no fault found" during the first testing pass is given by

$$s_f = \sum_{j=1}^{m} \left[\beta_j \prod_{l \neq j} (1-\alpha_l) \right] \rho_j \tag{3.7}$$

The probability that the diagnostics ultimately identify the subsystem that is in fact failed is given by

$$t_c = \frac{s_c}{1-s_f} \tag{3.8}$$

Suppose that the failed subsystem is subsystem j, $j \in \{1, 2, \ldots, m\}$. Let Δ_j denote the subsystem that is ultimately identified by the diagnostics as the failed subsystem. Consider the first pass through the diagnostics. Let $s_{j',j}$ denote the probability that subsystem j' is identified as the failed subsystem, $j' = 1, 2, \ldots, m$, and let $s_{nff,j}$ denote the probability that no fault is found. Then,

$$s_{j',j} = \begin{cases} \alpha_{j'} \prod_{l<j'} (1-\alpha_l) & \text{if } j' < j \\ (1-\beta_{j'}) \prod_{l<j'} (1-\alpha_l) & \text{if } j' = j \\ \alpha_{j'} \beta_j \prod_{\substack{l<j' \\ l \neq j}} (1-\alpha_l) & \text{if } j' > j \end{cases} \tag{3.9}$$

$j' = 1, 2, \ldots, m$, and

$$s_{nff,j} = 1 - \sum_{j'=1}^{m} s_{j',j} \tag{3.10}$$

Ultimately, the probability that subsystem j' is identified as the failed subsystem is given by

$$\Pr(\Delta_j = j') = \frac{s_{j',j}}{1 - s_{nff,j}} \tag{3.11}$$

$j' = 1, 2, \ldots, m$.

3.3 Evaluating Readiness

Once a subsystem is identified as failed, maintenance on the subsystem is initiated. Let Y_j denote the maintenance delay associated with subsystem j if it is actually the failed subsystem, and let Z_j denote the maintenance delay associated with subsystem j if it is not the failed subsystem, $j = 1, 2, \ldots, m$. If a subsystem is incorrectly identified as the failed subsystem, then after the delay it is not considered in the future iterations of the diagnostic process.

Unfortunately, the probability models associated with future iterations of the diagnostic process are too complex to evaluate analytically. Therefore, we utilize discrete-event simulation to estimate the readiness of a single system. This simulation model, constructed using the ARENA simulation software package, mimics the operation, failure, diagnosis and maintenance of a system over an extended period of time. As a result, this simulation model can be used to assess the impact of Type I and Type II errors on the readiness of the system. The model can also be used to explore the benefits of reducing some or all of the diagnostic error probabilities. Appendix A contains a thorough explanation of how to use the simulation model, including step-by-step instructions on how to input parameters and obtain the desired output average system readiness. Note that the simulation model itself is included on the CD accompanying this report. As a basis of reference, note that the average readiness of the system under perfect diagnostics would be given by

$$\frac{\frac{1}{\lambda}}{\frac{1}{\lambda}+\sum_{j=1}^{m}\left[E(Y_j)+\sum_{l=1}^{j}E(X_l)\right]\rho_j} \tag{3.12}$$

As an example, consider a system comprised of $m = 10$ subsystems having the characteristics enumerated in Table 3.1. The overall system failure rate is $\lambda = 0.002532$ per hour which implies a MTTF of 395 hours. In addition, the time required to perform diagnostics on a given subsystem is randomly selected from the range (0.05 hour, 0.1 hour). The maintenance delay associated with a correct diagnosis on a subsystem is drawn from a triangular probability distribution having a minimum of 2 days, a mode of 3 days and a maximum of 5 days. Also, the maintenance delay associated with an incorrect diagnosis on a subsystem is drawn from the triangular distribution having a minimum of 1 day, a mode of 1.5 days and a maximum of 2 days. Based on the specified model parameters, average system readiness under perfect diagnostics was determined to be 83.1%. However, for the case in which diagnostic errors are present, the average system readiness is 80.8%.

Table 3.1 System Parameters

j	λ_j *1000 hours	ρ_j	α_j	β_j
1	0.2808	0.1109	0.0798	0.0809
2	0.2797	0.1104	0.0508	0.0991
3	0.2693	0.1063	0.0707	0.0821
4	0.2629	0.1038	0.0777	0.0848
5	0.2538	0.1002	0.0735	0.0946
6	0.2445	0.0965	0.0559	0.0950
7	0.2427	0.0958	0.0541	0.0987
8	0.2410	0.0952	0.0527	0.0819
9	0.2333	0.0921	0.0567	0.0956
10	0.2244	0.0886	0.0644	0.0837

Now, consider the case of reducing the diagnostic error rates. Table 3.2 provides a summary of system performance resulting from several percent reductions in all Type I and Type II error probabilities. The results indicate that decreasing the error rates does not result in a significant increase in average system readiness.

Table 3.2 Results for Error Rate Reduction

% reduction in error rates	average system readiness
1	0.8082
2	0.8084
5	0.8082
10	0.8055
15	0.8118

Next, consider the case of reducing maintenance lead times. Tables 3.3 and 3.4 provide a summary of system performance resulting from several percent reductions in all maintenance lead times. The results show that as maintenance time is reduced the average system readiness increases significantly.

Table 3.3 Results for Reduction of Y_j

% reduction in Y_j	average system readiness
1	0.8093
2	0.8106
5	0.8145
10	0.8209
15	0.8279

Table 3.4 Results for Reduction of Z_j

% reduction in Z_j	average system readiness
1	0.8084
2	0.8085
5	0.8091
10	0.8102
15	0.8114

3.4 Evaluating Readiness for a Set of Systems

When multiple systems share the same diagnostic and maintenance resources, the assessment of the impact of diagnostics errors becomes more complex. Therefore, we extended our simulation model to consider this case. In addition to the mimicking the operation, failure, diagnosis and maintenance of each system, the model also includes: limitations of diagnostics test bed availability and spare parts inventory pools. For this model, let T denote the number of diagnostic test beds, and let V denote the time required to setup and prepare a given system for diagnostic testing. In addition let K_j denote the number of spare subsystems of type j available, and let L_j denote the time required to pull/install a subsystem of type j, $j = 1, 2, \dots, m$.

As an example, consider the scenario in which there are $q = 72$ systems each having the same characteristics as provided in the single system example. For this example let $V = 2.192$ hours. First, as a base case, consider the case in which there is one test bed, no spares, and L_j is randomly selected from the range (0.5 hours, 1.5 hours). Based on these specified model parameters, the average readiness is 74.9%.

Next, consider the case of increasing the number of test beds T. Table 3.5 summarizes the results of this analysis. It is clear that increasing the number test beds from one to two results in a significant increase in overall readiness. However, having more than three test beds results in a negligible improvement in average readiness.

Table 3.5 Results for Increasing T

T	average readiness
1	0.7491
2	0.7898
3	0.7912
4	0.7916
5	0.7919

Now consider the case of having $T = 3$ test beds and the addition of spares. The results of this analysis are provided in Table 3.6. As expected, the average readiness increases greatly as the spare inventory level is increased.

Table 3.6 Results with the Addition of Spares

K_j										average readiness
1	2	3	4	5	6	7	8	9	10	
1	0	0	0	0	0	0	0	0	0	0.7998
1	1	0	0	0	0	0	0	0	0	0.8068
1	1	1	0	0	0	0	0	0	0	0.8146
1	1	1	1	0	0	0	0	0	0	0.8219
1	1	1	1	1	0	0	0	0	0	0.8292
1	1	1	1	1	1	0	0	0	0	0.8357
1	1	1	1	1	1	1	0	0	0	0.8434
1	1	1	1	1	1	1	1	0	0	0.8504
1	1	1	1	1	1	1	1	1	0	0.8561
1	1	1	1	1	1	1	1	1	1	0.8617

3.5 Allocating Budgets for Diagnostic Improvement Efforts

Suppose a budget of $\$b$ has been allocated for improving the diagnostic capabilities relative to a single system, i.e. reducing Type I and Type II error probabilities. Furthermore, let $\gamma_j(\overline{\alpha}_j)$ denote the cost of reducing α_j to $\overline{\alpha}_j$, and let $\delta_j(\overline{\beta}_j)$ denote the cost of reducing β_j to $\overline{\beta}_j$, $j = 1, 2, \ldots, m$. In addition, the let the cost functions follow an exponential trend such as Figure 3.1 illustrates.

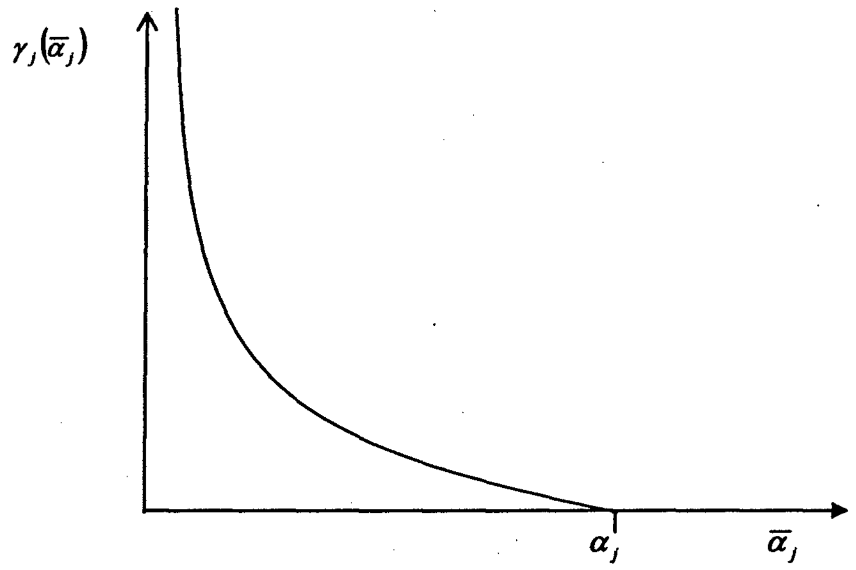


Figure 3.1 Cost Function Example

Since effective diagnostics can be characterized by "getting it right the first time", we can use the probability models developed previously in an optimization framework. The appropriate optimization model is given by

$$
\begin{array}{lll}
\max & s_c & \\
\text{subject to} & \sum_{j=1}^{m}\left[\gamma_j(\overline{\alpha}_j)+\delta_j(\overline{\beta}_j)\right]\le b & \\
 & 0<\overline{\alpha}_j\le\alpha_j & j=1,2,\ldots,m \\
 & 0<\overline{\beta}_j\le\beta_j & j=1,2,\ldots,m
\end{array}
$$

As an example, consider the same system characteristics as defined for the single system example. Next, the cost functions are given by

$$\gamma_j(\overline{\alpha}_j) = W_j\left[\frac{\alpha_j}{\overline{\alpha}_j} - 1\right] \tag{3.13}$$

and

$$\delta_j(\overline{\beta}_j) = G_j\left[\frac{\beta_j}{\overline{\beta}_j} - 1\right] \tag{3.14}$$

$j = 1, 2, \ldots, m$. Note that b = \$240,000 and the values of the cost function parameters are defined in Table 3.7.

Table 3.7 Cost Function Parameter Values

j	W_j	G_j
1	\$11,701	\$24,446
2	\$11,709	\$20,972
3	\$12,139	\$23,936
4	\$11,304	\$21,542
5	\$11,259	\$22,184
6	\$12,624	\$22,212
7	\$14,502	\$24,439
8	\$14,479	\$22,994
9	\$11,974	\$22,983
10	\$13,437	\$23,771

Before the optimization model was solved, a base case was calculated for comparison. For this case, the probability of correct diagnosis is found to be 0.689. Next, the optimization model is solved using Excel Solver. The optimized value of s_c was determined to be 0.847. The details of the optimal solution can be found in Table 3.8. The Excel spreadsheet used to perform the optimization is included on the CD accompanying this report.

Table 3.8 Results of Optimization Model Example

j	α_j	$\overline{\alpha}_j^*$	β_j	$\overline{\beta}_j^*$
1	0.0798	0.0175	0.0809	0.0663
2	0.0508	0.0149	0.0991	0.0735
3	0.0707	0.0192	0.0821	0.0707
4	0.0777	0.0213	0.0848	0.0655
5	0.0735	0.0228	0.0946	0.0765
6	0.0559	0.0239	0.0950	0.0743
7	0.0541	0.0299	0.0987	0.0794
8	0.0527	0.0368	0.0819	0.0750
9	0.0567	0.0489	0.0956	0.0859
10	0.0644	0.0644	0.0837	0.0822

4. A Brief Study of Prognostics

Consider a system that can be represented by a single, "black box" component. A new copy of this component has a Weibull time to failure distribution having shape parameter $\beta > 1$ and scale parameter $\eta > 0$. Note that the fact that $\beta > 1$ implies that the component has an increasing failure rate. The system is required to perform a sequence of missions having length m. If the system fails during a mission, then the mission is aborted and maintenance is performed. The time required to perform maintenance is t_m, and maintenance restores the system to an "as good as new" condition. The performance of this system is measured using $E(S)$, the average time between successful missions. Note that if the system never failed, then $E(S)$ would be equal to m.

We use a discrete-event simulation model to assess the performance of this system under three maintenance policies. The first policy is referred to as "run-to-failure" (RTF). Under this policy, the system is maintained only upon failure. The second policy is referred to as "preventive maintenance" (PM). Under this policy, an optimal, scheduled, preventive maintenance policy is applied to the system. This policy is summarized by the parameter τ. If the system successfully completes τ consecutive missions without maintenance, then maintenance is performed prior to the next mission. Note that the optimal value of τ is estimated using the simulation model. Clearly, system performance under RTF is no better than system performance under PM, and performance under either policy is inferior to a "perfect system" (no failures).

The third maintenance policy considered is referred to as "prognostics". Under this policy, the remaining life of the system is estimated at the end of each successful mission. If this estimate is less than the mission length, then maintenance is performed prior to the next mission. We first consider "perfect prognostics", i.e. the estimate of remaining life is exactly equal to the remaining life. System performance under perfect prognostics is at least as good as system performance under PM. However, perfect prognostics is an unrealistic standard. Therefore, we also consider the case in which the estimate of remaining life of the system is subject to error. We model this error as a normal random variable having a mean of zero and a standard deviation of α. This error creates the possibility of unnecessarily early maintenance and system failures due to overestimated remaining life.

Figure 4.1 contains a plot of the type of analysis facilitated by the simulation model. This plot contains the results of twelve experiments (E1, E2, ... , E12) having a common value of m = 0.25 and $\eta = 1$ (β and t_m vary). Note that the last five items on the legend correspond to perfect prognostics and four increasing levels of prognostic errors. In some cases, e.g. E1, PM is no better than RTF. In other cases, e.g. E2, the worst prognostics errors still provide system performance at least as good as that under PM. Finally, some cases, e.g. E12, indicate that small prognostic errors can result in system performance that is worse than that under RTF.

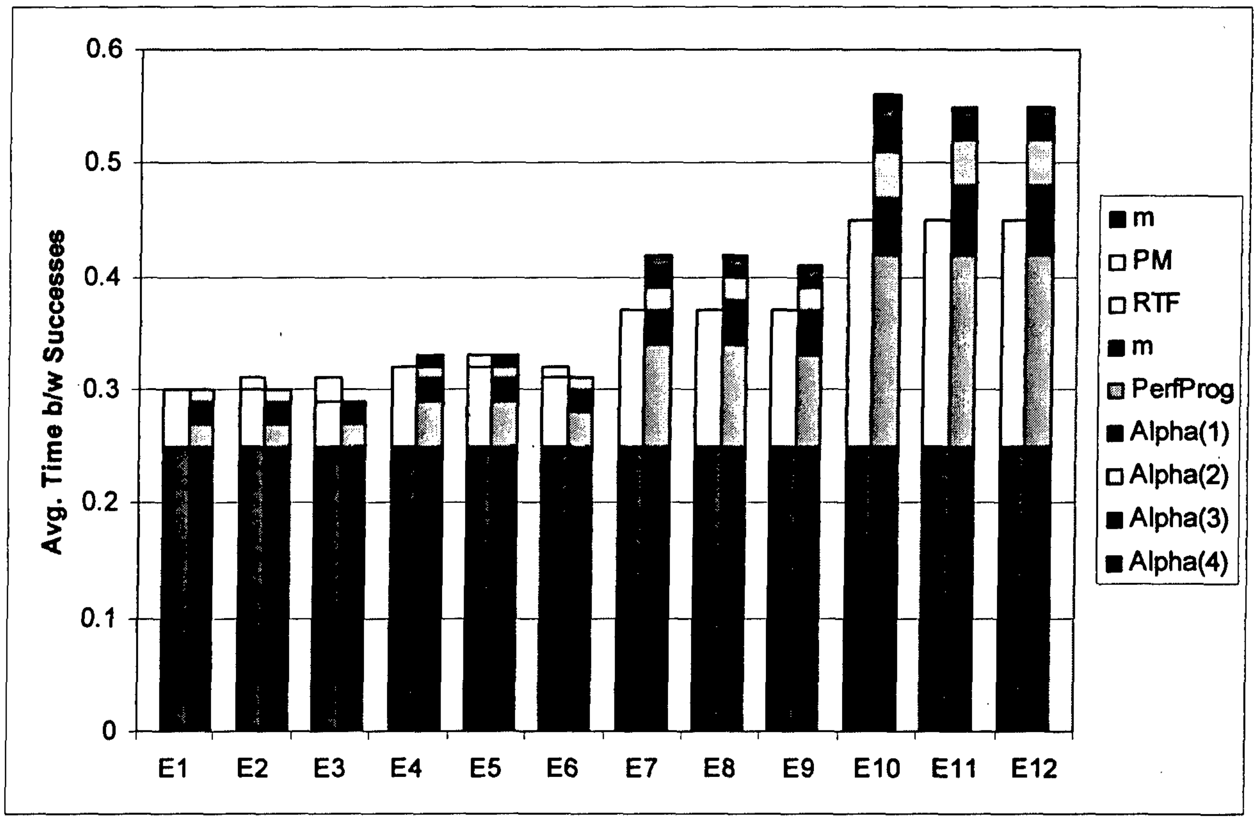


Figure 4.1 Sample Simulation Results

References

[1] N. A. E. Aly and A. A. Aly, "Measures of Testability for Automatic Diagnostic Systems," *IEEE Transactions on Reliability*, vol 37, no 5, 1988, pp 531-538.

[2] S. Deb, K. R. Pattipati, V. Raghavan, M. Shakeri and R. Shrestha, "Multi-Signal Flow Graphs: A novel approach for System Testability Analysis and Fault Diagnosis," *IEEE AES Systems Magazine*, vol 10, no 5, 1995, pp 14-25.

[3] W. G. Fenton, T. M. McGinnity and L. P. Maguire, "Fault Diagnosis of Electronic Systems Using Intelligent Techniques: A Review," *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, vol 31, no 3, 2001, pp 269-281.

[4] J. Guan and J. H. Graham, "Diagnostic Reasoning with Fault Propagation Digraph and Sequential Testing," *IEEE Transactions on Systems, Man and Cybernetics*, vol 24, no 10, 1994, pp 1552-1558.

[5] A. A. AL-Jumah and T. Arslan, "Artificial Neural Network Based Multiple Fault Diagnosis in Digital Circuits," *Proceedings ICCAS*, vol 2, 1998, pp 304-307.

[6] A. Mathur, K. F. Cavanaugh, K. R. Pattipati, P. K. Wilett and T. R. Galie, "Reasoning and Modeling Systems in Diagnosis and Prognosis," *Proceedings of the 2001 SPIE*, vol 4389, 2001, pp 194-203.

[7] J. H. Murphy and B. J. Kagle, "Neural Network Recognition of Electronic Malfunctions," *Journal of Intelligent Manufacturing*, 1992, pp 205-216.

[8] J. A. Nachlas, S. R. Loney and B. A. Binney, "Diagnostic-strategy selection for series systems," *IEEE Transactions on Reliability*, vol 39, no 3, 1990, pp 273-280.

[9] K. R. Pattipati and M. G. Alexandridis, "Application of Heuristic Search and Information Theory to Sequential Fault Diagnosis," *IEEE Transactions on Systems, Man and Cybernetics*, vol 20, no 4, 1990, pp 872-887.

[10] K. R. Pattipati and M. Dontamsetty, "On a Generalized Test Sequencing Problem," *IEEE Transactions on Systems, Man and Cybernetics*, vol 22, no 2, 1992, pp 392-396.

[11] V. Raghavan, M. Shakeri and K. R. Pattipati, "Test Sequencing Algorithms with Unreliable Tests," *IEEE Transactions on Systems, Man and Cybernetics - Part A: Systems and Humans*, vol 29, 1999, pp 347-357.

[12] M. J. Roemer, G. J. Kacprzynski and M. H. Schoeller, "Improved Diagnostic and Prognostic Assessments Using Health Management Information Fusion," *AUTOTESTCON (Proceedings)*, 2001, pp 365-377.

[13] M. Shakeri, V. Raghavan, K. R. Pattipati, A. Patterson-Hine, "Sequential Testing Algorithms for Multiple Fault Diagnosis," *IEEE Transactions on Systems, Man and Cybernetics - Part A: Systems and Humans*, vol 30, no 1, 2000, pp 1-14.

[14] K. A. E. Totton and P. R. Limb, "Experience in Using Neural Networks for Electronic Diagnosis," *Proceedings of 2nd International Conference on Artificial Neural Networks*, 1991, pp 115-118.

[15] G. Vachtsevanos, "Performance Metrics for Fault Prognosis of Complex Systems," *AUTOTESTCON (Proceedings)*, 2003, pp 341-345.

[16] H. Wu, Y. Liu and Y. Ding, "A Method of Aircraft Unit Fault Diagnosis," *Aircraft Engineering and Aerospace Technology*, vol 75, no 1, 2003, pp 27-32.

[17] J. Ying, T. Kirubarajan, K. R. Pattipati and A. Patterson-Hine, "A Hidden Markov Model-Based Algorithm for Fault Diagnosis with Partial and Imperfect Tests," *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, vol 30, no 4, 2000, pp 463-473.

Appendix A – Arena Simulation Model

(Screen shots see Figures A.1 through A.12)

Open the Arena file named *evaluating system readiness*

To modify input parameters:

1. Click on the Basic Process Panel tab and then open the variable data module.

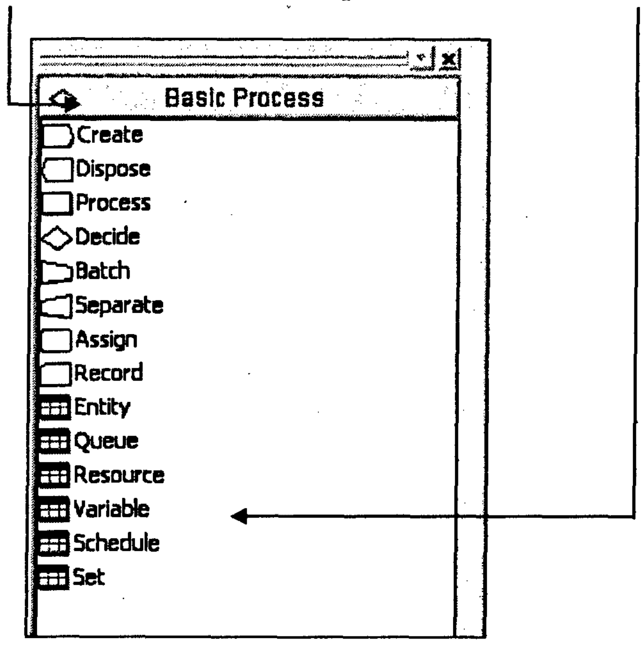
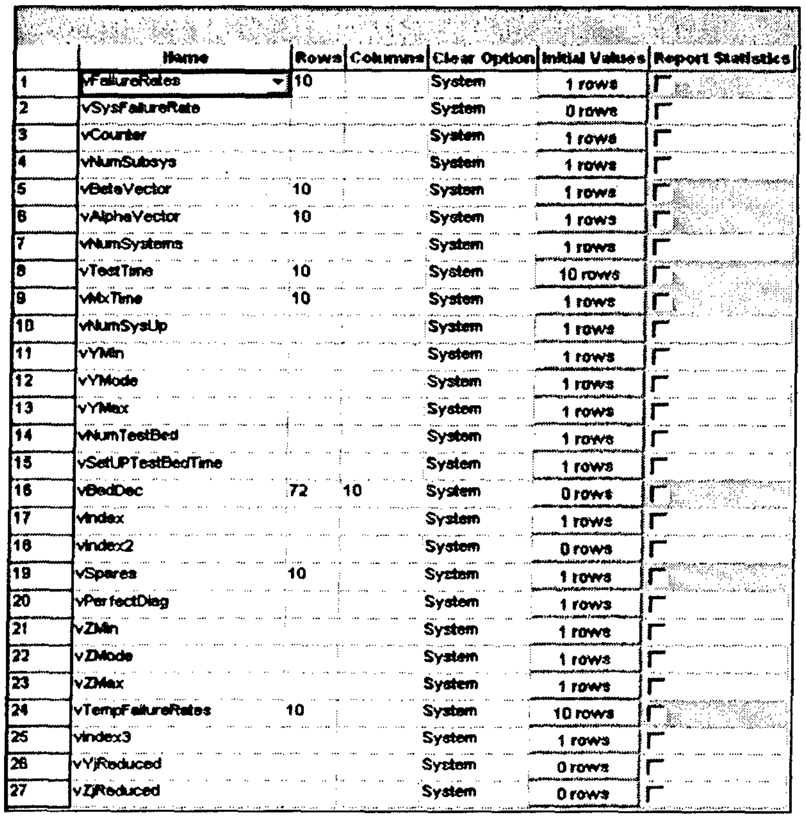


Figure A.1Basic Process Panel

2. Once the Variable Data Module has been opened, a list of 27 variables will be displayed at the bottom of the screen.



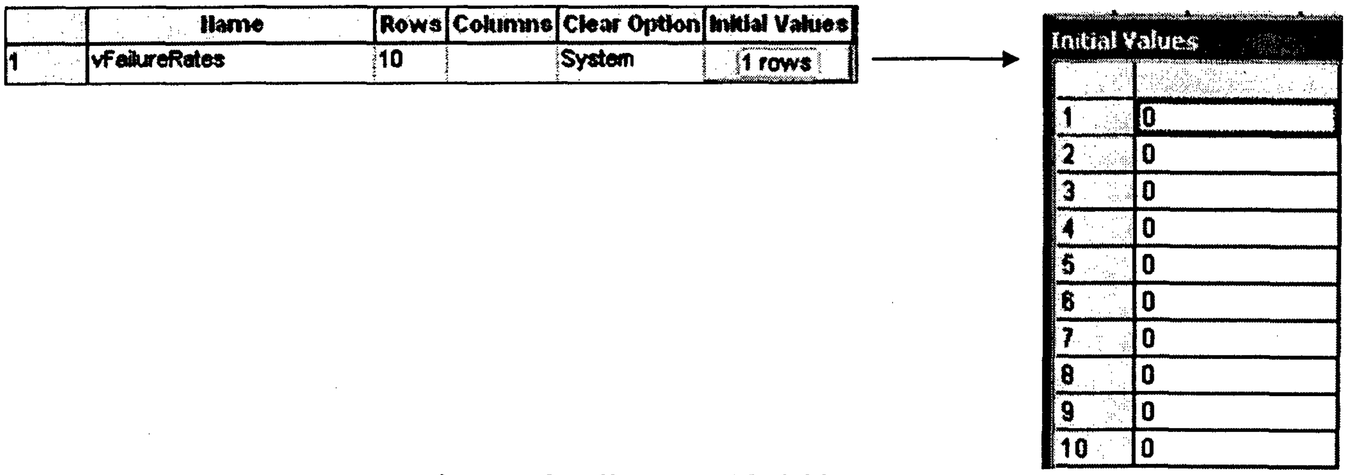
	Name	Rows	Columns	Clear Option	Initial Values	Report Statistics
1	vFailureRates	10		System	1 rows	☐
2	vSysFailureRate			System	0 rows	☐
3	vCounter			System	1 rows	☐
4	vNumSubsys			System	1 rows	☐
5	vBetaVector	10		System	1 rows	☐
6	vAlphaVector	10		System	1 rows	☐
7	vNumSystems			System	1 rows	☐
8	vTestTime	10		System	10 rows	☐
9	vMxTime	10		System	1 rows	☐
10	vNumSysUp			System	1 rows	☐
11	vYMin			System	1 rows	☐
12	vYMode			System	1 rows	☐
13	vYMax			System	1 rows	☐
14	vNumTestBed			System	1 rows	☐
15	vSetUPTestBedTime			System	1 rows	☐
16	vBedDec	72	10	System	0 rows	☐
17	vIndex			System	1 rows	☐
18	vIndex2			System	0 rows	☐
19	vSpares	10		System	1 rows	☐
20	vPerfectDiag			System	1 rows	☐
21	vZMin			System	1 rows	☐
22	vZMode			System	1 rows	☐
23	vZMax			System	1 rows	☐
24	vTempFailureRates	10		System	10 rows	☐
25	vIndex3			System	1 rows	☐
26	vYjReduced			System	0 rows	☐
27	vZjReduced			System	0 rows	☐

Figure A.2 Variable Data Module

3. These variables allow system readiness to be evaluated under different parameters. However, it must be noted that not all variables are meant to be changed by the user and the ones eligible for user change must stay within certain limits in order for the simulation model to run.

The following defines each variable, its limits and default values:

vFailureRates – This variable is an array of ten rows in which the failure rates for each subsystem will be stored. The default values of this array must always equal zero.



	Name	Rows	Columns	Clear Option	Initial Values
1	vFailureRates	10		System	1 rows

Initial Values	
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0

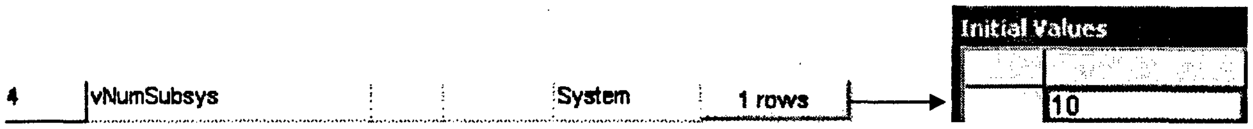
Figure A.3 Failure Rate Variable

To view or change the initial value of the variable array right click on the box titled "1 row" underneath Initial Values and an input box will appear as shown.

vSysFailureRate – This variable represents the system failure rate λ. Its initial value must also always equal zero.

vCounter – This is a counter variable used within the logic and its initial value must always be equal to one.

vNumSubsys – Declares the number of subsystems within each system.

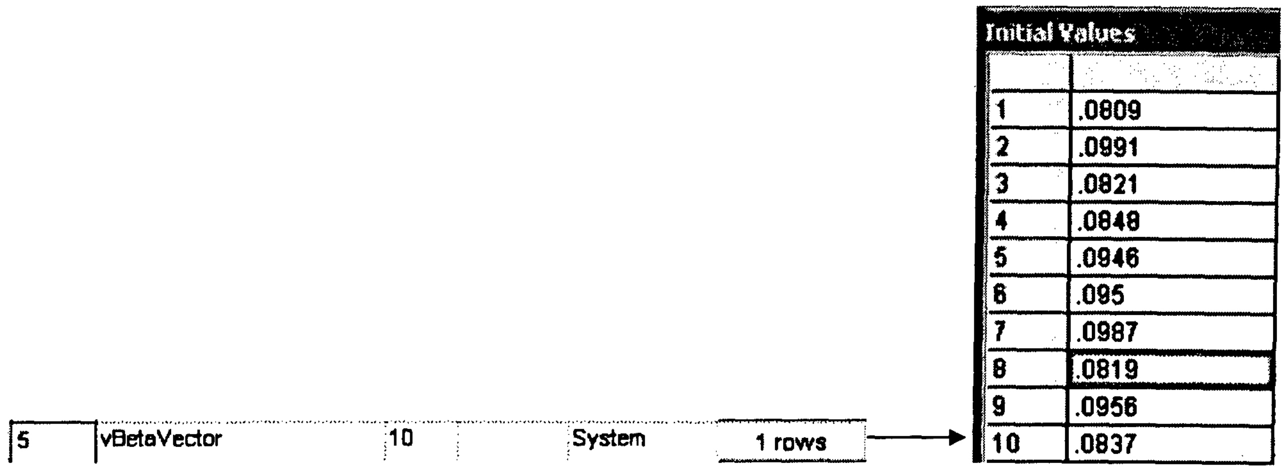


4	vNumSubsys			System	1 rows

Initial Values	
	10

Figure A.4 Number of Systems Variable

vBetaVector – Specifies the probability of a Type II Error (false negative) for each subsystem from $j = 1$ to m - β_j.



5	vBetaVector	10	System	1 rows

Initial Values

1	.0809
2	.0991
3	.0821
4	.0848
5	.0946
6	.095
7	.0987
8	.0819
9	.0956
10	.0837

Figure A.5 Beta Vector Variable

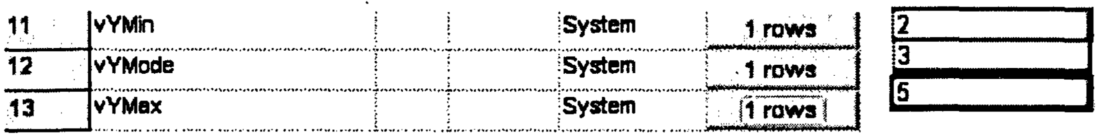
vAlphaVector – Specifies the probability of a Type I Error (false positive) for each subsystem from $j = 1$ to m - α_j

vNumSystems – Represents the number of q independent and identical systems. In order for the model to run this number must be more than or equal to 1 and less than or equal to 72. In other words, we must have a least one system but the maximum we can have is 72.

vMxTime – This is a ten row array storing the maintenance time L_j associated with each subsystem. It denotes the time required to pull/install a subsystem of type j.

vNumSysUp – Used to store the number of systems up at any given time. Its initial value must be left at 0.

vYMin, vYMode, and vYMax - Denotes the maintenance delay *Yj* associated with correctly identifying subsystem *j* as the failed subsystem. This delay time is drawn from a triangular probability distribution having a minimum of 2 days, a mode of 3 days and a maximum of 5 days. Hence, within the model three variables were created to input the minimum, mode, and maximum of this distribution. The initial values are 2, 3, and 5 but may be changed.



11	vYMin			System	1 rows	2
12	vYMode			System	1 rows	3
13	vYMax			System	1 rows	5

Figure A.6 Maintenance Delay Variable (*Yj*)

vNumTestBed – Denotes the number of diagnostic test beds T_j available. Its initial value must be more than or equal to one since the model needs at least one test bed to perform diagnostic testing.

vSetUpTestBedTime – Denotes the time *V* needed to setup and prepare system for diagnostic testing. Its initial value is set to 2.2192 hours but it may be changed.

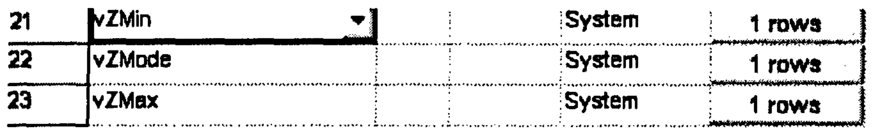
vBadDec – Is a matrix containing 72 rows and 10 columns. This matrix is used to store a binary variable that is used within the logic of the model to keep track of whether you have incorrectly identified a failed subsystem. If a subsystem is incorrectly identified as the failed subsystem, then after the delay, it is not considered in future iterations of the diagnostic process.

vIndex and vIndex2 – This is also a counter variable used in the logic of the model. Its initial value must be zero.

vSpares - Denotes the number of spare subsystems K_j.

vPerfectDiag – Used to indicate whether the model will run under perfect diagnostics. If 1 is used that means all diagnostic decisions will be correct. If 0 is used there will be errors associated with the decisions.

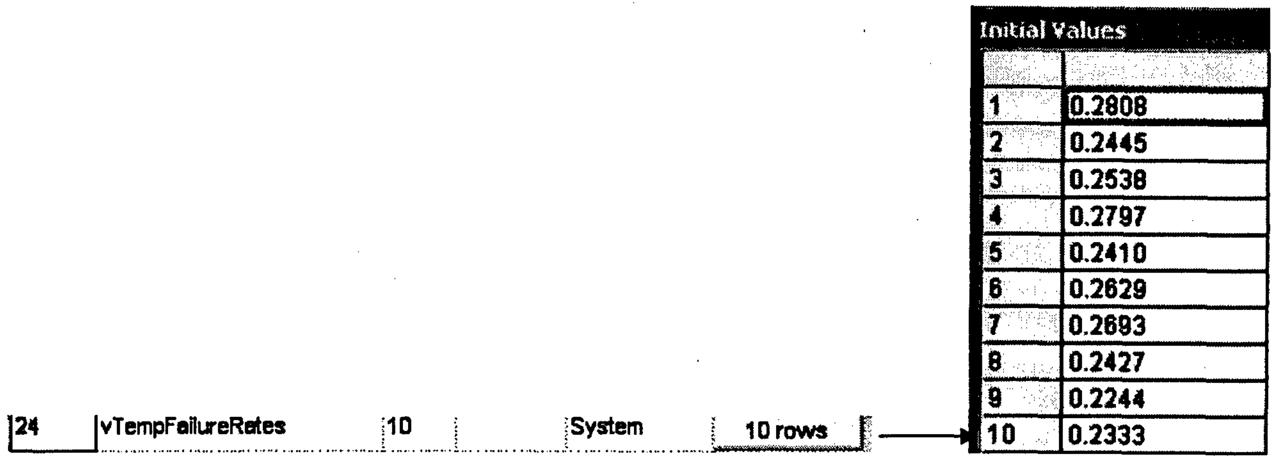
vZMin, vZMode, and vZMax – Denotes the maintenance delay Z_j associated with incorrectly identifying subsystem j as the failed subsystem. This delay time is drawn from a triangular probability distribution having a minimum of 1 day, a mode of 1.5 days and a maximum of 2 days. Hence, within the model three variables were created to input the minimum, mode, and maximum of this distribution. The initial values are 1, 1.5, and 2 but may be changed.



21	vZMin			System	1 rows
22	vZMode			System	1 rows
23	vZMax			System	1 rows

Figure A.7 Maintenance Delay Variable (Z_j)

vTempFailureRates - This matrix consisting of ten rows is used to input the failure rate λ_j associated with each subsystem. These failure rates are based in failures per 1000 hours.



24	vTempFailureRates	10	System	10 rows

Initial Values

1	0.2808
2	0.2445
3	0.2538
4	0.2797
5	0.2410
6	0.2629
7	0.2693
8	0.2427
9	0.2244
10	0.2333

Figure A.8 Failure Rate Variable

vIndex3 – Used as a counter variable. Its initial value must be set equal to one.

vYjReduced and vZjReduced – These variables are used to reduce the maintenance delays. For example, if maintenance delay Y_j is to be reduced by 15%, 0.15 must be entered for Y_j.

Once all variables have been checked and entered the model is ready to run. To initialize the model you can go to Run and then select Go (Figure A.9) or you can click on the run button ▶ located in the Standard Toolbar.

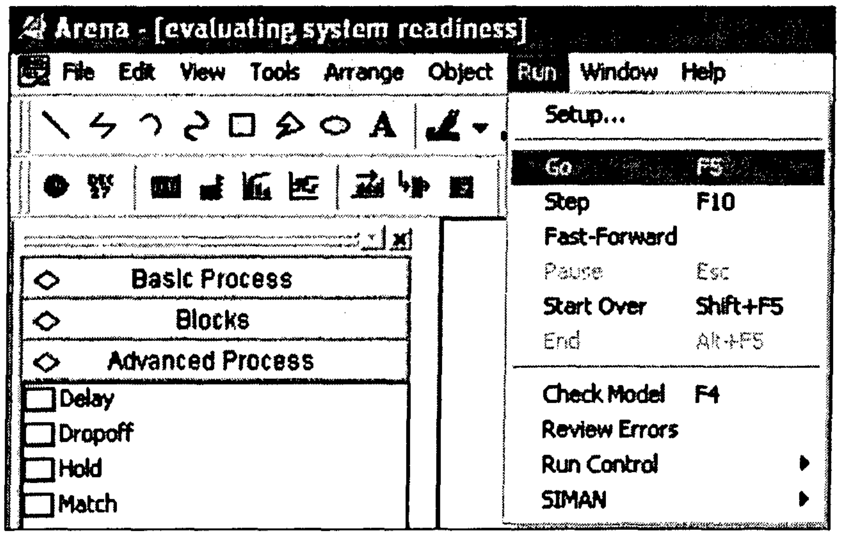
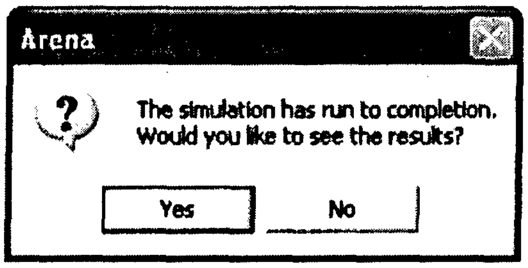


Figure A.9 Run Setup Menu

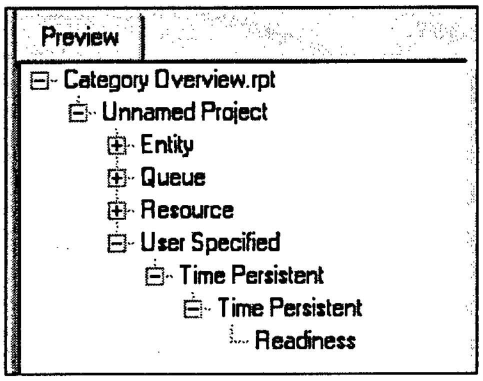
Once the model is finished running a window will appear asking if you want to see the results (Figure A.10).
Click Yes.


Arena

The simulation has run to completion. Would you like to see the results?

Figure A.10 Simulation Completion Window

The Category Overview Results Screen will then appear. Click on Category Overview, then on User Specified and finally on the time persistent variable Readiness.

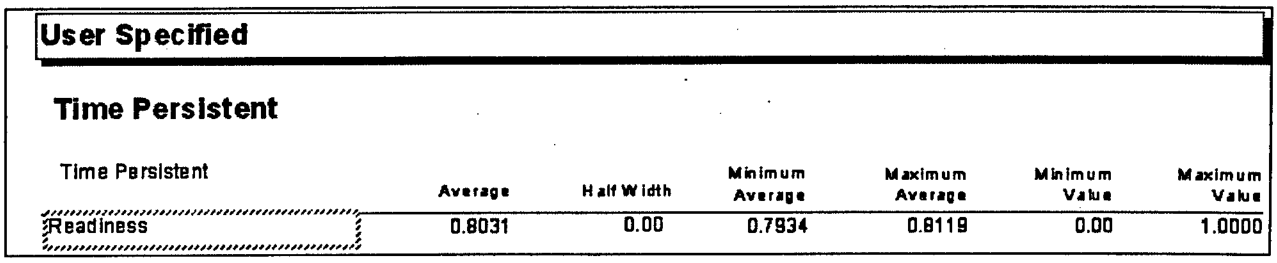


Preview

- Category Overview.rpt
  - Unnamed Project
    - Entity
    - Queue
    - Resource
    - User Specified
      - Time Persistent
        - Time Persistent
          - Readiness

Figure A.11 Preview Window

The report window will show the resulting average readiness value.



User Specified

Time Persistent

Time Persistent	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Readiness	0.8031	0.00	0.7934	0.8119	0.00	1.0000

Figure A.12 Readiness Report